Санкт-Петербургский государственный университет

Кафедра технологии программирования

Санникова Елена Сергеевна

Кластеризация документов в системе документооборота и документационного управления

Выпускная квалификационная работа бакалавра

Научный руководитель:
к. ф.-м. н. Добрынин В. Ю.

Рецензент:
руководитель научной лаборатории Digital Design Ашихмин И. А.

Санкт-Петербург
2017

Оглавление

Введение

Неотъемлемой частью каждой современной компании является система документооборота. Организация работы с внешними и внутренними документами является довольно трудоемкой и ресурсозатратной деятельностью предприятий, из-за чего она может стать узким горлышком любого хорошо организованного бизнес-процесса. В динамично развивающемся мире электронных технологий процесс обмена документами также необходимо переводить на новый уровень развития. На помощь приходят электронные решения, позволяющие упростить и автоматизировать данную часть бизнес-процесса. Последнее время переходу на электронный документооборот хорошо способствует расширение применения электронной подписи, а так же повышенное внимание к вопросам информационной безопасности.

Появление и внедрение современных систем электронного документооборота позволило сократить временные затраты, обеспечить прозрачность документопотока, а также бизнес-процессов, повысить дисциплину сотрудников к исполнению задач, фиксации результатов, защитить информацию от потерь, утечек, обеспечить безопасность данных и строгое разграничение прав доступа.

Из-за достаточно большого количества преимуществ системы электронного документооборота начинают внедряться повсеместно. Как следствие, компании накапливают достаточно большой объем электронных документов, который может позволить найти общие закономерности данных, интересные зависимости, новые характеристики, а так же построить модели данных путем применения алгоритмов машинного обучения и анализа данных.

Одной из задач анализа данных, которые получены из системы электронного документооборота какой-либо компании, можно рассмотреть задачу построения кластерной модели данных. Кластерный анализ [13] представляет из себя задачу разбиения исходного набора элементов выборки на непересекающиеся множества, называемые кластерами, так, чтобы кластер содержал в себе наиболее похожие элементы, а элементы

двух различных кластеров существенно отличались. Кластерный анализ данных применяется давно и имеет широкий круг применимости. С его помощью можно решить такие задачи, как сжатие данных, сократив исходную избыточную выборку, заменяя каждый кластер на один наиболее представительный для него элемент. Можно решать задачу поиска новизны, когда новый элемент не удается присоединить ни к одному из кластеров. А также можно использовать довольно важную, на мой взгляд, стратегию "разделяй и властвуй", которая заключается в том, что после получения кластерной структуры данных мы можем упростить дальнейшую работу с данными, применяя методы анализа и машинного обучения к каждому кластеру отдельно. Такой подход позволяет эффективнее работать с большими объемами данных, а так же получать более точные результаты.

Если учесть объемы документооборота, которыми обладают даже средние компании, то задача разбиения этого объема документов на кластеры для дальнейшего анализа становится весьма актуальной. Поэтому в данной работе будут рассмотрены подходы к построению кластерной структуры для коллекции документов из системы электронного документооборота.

Постановка задачи

Одной из систем электронного документооборота является система Docsvision [16], разработанная компанией Digital Design [3]. Компания предоставила мне возможность заняться решением задачи кластеризации документов. Одним из пользователей системы Docsvision с 2009 года является Правительство Мурманской области. Основная деятельность Правительства неразрывно связана с работой с внутренними и внешними документами и поручениями. И от качества и эффективности документационного взаимодействия во многом зависит оперативность и эффективность работы органов государственной власти. Поэтому компания постоянно совершенствует систему и стремится улучшать и автоматизировать еще больше процессов, происходящих в документообороте.

Глобальная задача для нас, студентов, проходящих практику в компании, была следующая. Каждый документ в системе Docsvision, состоит не только из текстового файла, который в компании считается документом, но и некоторой "карточки" с атрибутами этого документа. Некоторые атрибуты появляются при регистрации документа в системе, другие - в процессе работы над документом сотрудниками компании. Нас попросили научиться предсказывать эти атрибуты различными способами. К примеру, по загруженному в систему файлу предсказать атрибуты, которые введет для этого документа регистратор. Или у нас имеются почти все атрибуты и текстовый файл, и нам необходимо предсказать оставшиеся атрибуты. Решение этой задачи существенно упростила бы работу регистраторов документов.

Естественно, что объем накопленных документов в Правительстве с 2009 года является довольно большим, в следствии чего возникает потребность для решения поставленной задачи сперва применить на этих данных стратегию "разделяй и властвуй". Т.е. для более точного предсказания атрибутов документа будет лучше строить алгоритм предсказания для каждого набора похожих документов отдельно. Поэтому построение кластерной структуры данных стало одной из задач

на пути к решению глобальной, и именно ее решением я занималась.

Для ее решения были поставлены следующие задачи:

1. Провести анализ данных и преобразовать их так, чтобы с ними могли работать различные алгоритмы кластеризации. В особенности это касается текстовых файлов и категориальных атрибутов.

2. Протестировать на преобразованных данных различные алгоритмы кластеризации.

3. Численно оценить качество работы алгоритмов и их вариаций.

4. Учитывая полученные оценки, выбрать лучшее решение для поставленной задачи.

Обзор литературы

Для погружения в область кластеризации данных прекрасно подходит книга "Algorithms for Clustering Data" [1]. В ней описаны все стандартные алгоритмы кластеризации, различные способы предобработки и проекции данных, в частности есть подробное описание метода многомерного шкалирования (multidimensional scaling), который использовался мной в данной работе.

С задачей построения тематической модели корпуса документов, а именно с алгоритмом Latent Dirichlet Allocation (LDA) помогла разобраться одноименная статья Дэвида Блея (David Blei) [2], опубликованная в 2003 году. Так же помогает хорошо разобраться в алгоритме, а особенно в его математической основе статья "Parameter estimation for text analysis" [5], опубликованная в 2008 году.

Для интерактивной визуализации результатов построенной тематической модели использовалась Python-библиотека pyLDAvis [11]. Для того, чтобы понять, каким образом происходит отображение на графике множества тем, а так же распределений слов в темах, полезно ознакомиться со статьей "LDAvis: A method for visualizing and interpreting topics" [8] от разработчиков данной библиотеки.

В работе использовались различные алгоритмы на графах: построение остовного дерева, нахождение эйлерова цикла, приближенные алгоритмы решения задачи комивояжера. Описания алгоритмов можно найти в книге "Approximation algorithms" [10, p. 27-37].

1. Подготовка данных

1.1. Описание данных

При регистрации документа в системе Docsvision, оператор заполняет форму, в которой указывает некоторый набор параметров, присущих документу, и прикрепляет один или несколько файлов. Пример такой формы представлен на рис. 1. Введенные параметры с некоторым набором параметров, которые заполняются системой автоматически, образуют набор характеристик документа и в совокупности называются "карточкой" документа. Отдельным документом в системе Docsvision является совокупность из "карточки" и набора прикрепленных при регистрации файлов.

Мне был предоставлен корпус из 131214 документов Правительства Мурманской области. Каждый документ представлял из себя отдельную директорию, в которой находились "карточка" в формате JSON, и относящиеся к этому документу файлы.

1.2. Выборка информативных атрибутов документа

Чтобы определить, какими атрибутами обладают документы из корпуса, были проанализированы карточки документов. Карточка представляет из себя документ в формате JSON. Его сокращенный вариант представлен в Listing 1. Каждый документ обладает своим уникальным идентификатором и набором атрибутов. Из этого набора были выделены следующие:

1. "AccessType": определяет, каким уровнем доступа обладает документ. Всего в корпусе представлено только 3 варианта этого атрибута: "ДСП", "Конфиденциально" и "Общий".

2. "Kind": определяет тип документа. Например: "Закон", "Приказ", "Запрос", "Протокол". Всего представлено 19 различных типов в корпусе.

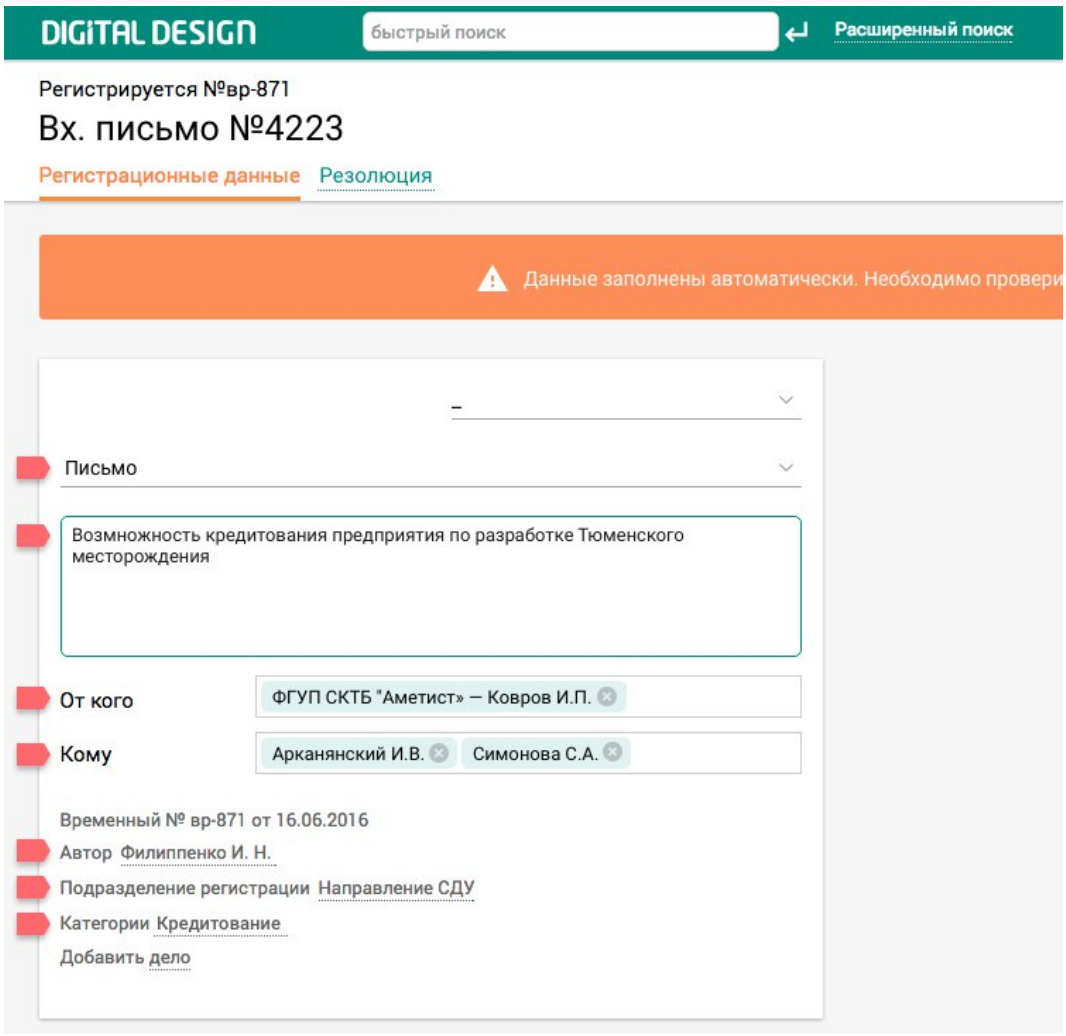

Рис. 1: Пример формы для регистрации документа в СЭД

3. ”Operator”: содержит идентификатор человека, которым был создан документ в системе. Всего представлено 145 уникальных идентификаторов.

4. ”Registrator”: содержит идентификатор человека, который зарегистрировал документ в системе. Всего представлено 158 уникальных идентификаторов.

5. ”RegistratorDepartment”: содержит идентификатор департамента, в котором был зарегистрирован документ. Всего представлено 66 уникальных идентификаторов департаментов.

6. ”Addressees”: представляет из себя список идентификаторов адресатов. Тип адресата можно узнать из поля ”AddresseeType”, при-

Таблица 1: Атрибуты документа

Имя	**Описание**	**Количество уникальных значений**	**Тип**
AccessType	Тип доступа	3	Id
Kind	Тип документа	19	Id
Operator	Создал	145	Id
Registrator	Подтвердил	158	Id
RegistratorDepartment	Подразделение регистрации	66	Id
Sender	Отправитель	4339	Id
Recipients	Получатели	634	list of Id
Categories	Категории	157	list of Id

сущего для каждого адресата. Если поле "AddresseeType" равно 1, то это отправитель, если 0 или 2, то это получатель или фактический получатель. Уникальных идентификаторов отправителей оказалось довольно много (4339 штук), и большинство из них встречалось только в одном документе, а вот получателей оказалось 634.

7. "Categories": представляет из себя список идентификаторов категорий, к которым можно отнести данный документ. Примеры категорий: "Финансы", "Выборы, избирательная система", "Система образования", "Коммунальное хозяйство". Всего представлено 157 различных категорий в корпусе.

Все эти атрибуты являются категориальными. В Таблице 1 представлена краткая информация по выбранным атрибутам.

1.3. Информация из текстовых документов

Файлы, прикрепленные при регистрации документа, могут быть в различных форматах: doc, docx, txt, pdf, tiff. Основная масса файлов в предоставленном корпусе имела формат tiff, который представляет из себя сканированные растровые изображения бумажного документа. Если из остальных представленных форматов извлечение текста не составляет проблем, то для tiff-файлов необходимо было использовать инструменты распознавания текста. Для этой задачи была выбрана

Listing 1: Сокращенный вариант Document.json

```
{
  ''Id'': ''ab1af2e6-646f-4764-ab02-4bfef4633720'',
  ''AccessType'': {
    ''Name'': ''Конфиденциально'',
    ''Id'': ''feb9ad69-a678-47d1-9d68-9882bde9d3bd''
  },
  ''Kind'': {
    ''Name'': ''Приказ'',
    ''Id'': ''a11c6bb7-6e16-485c-9668-9aacffd60d98''
  },
  ''Operator'': {
    ''Id'': ''660a2ae3-5264-4898-ab78-1515c5f10cb1''
  },
  ''Registrator'': {
    ''Id'': ''660a2ae3-5264-4898-ab78-1515c5f10cb1''
  },
  ''RegistratorDepartment'': {
    ''Id'': ''d405190c-9f1c-49f0-9ebe-817d30207d18''
  },
  ''Addressees'': [
    {
      ''Reference'': {
        ''Id'': ''eac1b627-9d22-4960-a427-b23decbd3bde''
      },
      ''AddresseeType'': 0,
    },
    ...
  ],
  ''Categories'': [
    {
      ''Name'': ''Финансы'',
      ''Id'': ''e953bc4d-2d5b-4d12-a716-b13fcf8e2c87''
    },
    ...
  ],
}
```

свободная компьютерная программа Tesseract [9], предназначенная для оптического распознавания рукописного или печатного текста.

В итоге у нас каждый документ характеризуется некоторым текстом. Что мы из этого можем получить? Почему бы нам не извлечь из текстов такую важную информацию, как темы, которые в нем описываются? Если получится это сделать, то можно было бы использовать список тем как еще один дополнительный атрибут. Да, у нас есть уже атрибут "Categories", который как раз описывает, к каким темам относится документ. Но при анализе этого атрибута было обнаружено, что только одной трети документам присвоена хотя бы одна категория. Для остальных документов значение этого атрибута оказалось пустым. Кроме того, при просмотре имен этих 157 уникальных значений атрибута можно заметить, что среди них много дубликатов. Не очень хорошее качество данного атрибута еще сильнее побуждает реализовать идею извлечения тем для документа из текстов.

Для решения этой задачи использовалась генеративная вероятностная тематическая модель LDA (latent Dirichlet allocation). Основная идея модели заключается в том, что документ не привязан жестко только к одной теме, а представляет собой смесь из тем, а каждое слово порождено одной из тем этой смеси. Т.е. тема представляет из себя распределение вероятностей на словах, а документ - распределение вероятностей на темах. Основой модели является предположение, что распределение вероятностей принадлежности тем к документам является случайным вектором из распределения Дирихле с параметром α.

Перед построением тематической модели тексты были предобработаны при помощи библиотек языка Python. С помощью RegexpTokenizer из библиотеки nltk [6] были выделены русские слова, из них были удалены все стоп-слова и слова с длинной меньше четырех символов. Далее с помощью библиотеки pymorphy2 [14] оставшиеся слова были приведены к нормальной форме. После такой предобработки каждый документ представляет из себя список слов. Такие данные уже можно передавать на вход алгоритму для построения тематической модели. Модель LDA была построена с помощью библиотеки Gensim [4]. На вход алгоритм

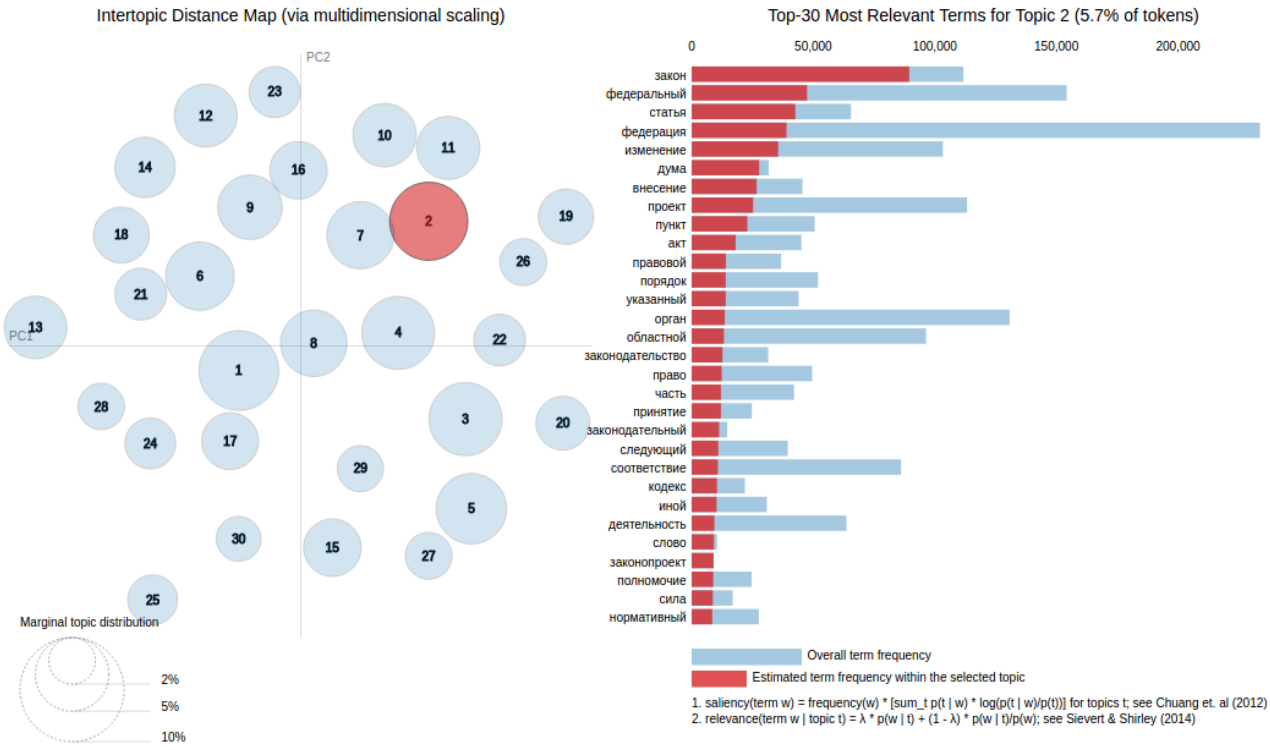

Рис. 2: Визуализация модели LDA при помощи библиотеки PyLDAvis

требует ввести количество тем, для которого мы хотим построить модель. Нами было построено несколько моделей, а именно модели для 20, 25, 30, 35, и 40 тем. Результаты были визуализированы при помощи библиотеки pyLDAvis [11], которая позволяет графически изобразить распределения слов по темам. Пример визуализации одной из моделей представлен на рис. 2. Слева на плоскости располагаются темы так, что чем больше похожи распределения слов для двух тем, тем ближе темы располагаются друг к другу на плоскости. Выбирая одну из тем, справа можно увидеть топ-30 слов, которые эту тему характеризуют. Пять таких визуализаций, по одной для каждой нашей модели, были проанализированы несколькими коллегами из Digital Design на предмет интерпретируемости человеком полученных тем. Модель из 30 тем всеми была оценена как лучшая, поэтому в дальнейшей работе использовалась именно она. Топ-10 слов для первых 10 наиболее популярных в корпусе тем этой модели представлены в Таблице 2.

Теперь, имея тематическую модель нашего корпуса документов, мы можем для любого документа получить вектор вероятностей распреде-

ления тем и использовать его как еще один атрибут для нашего корпуса.

Таблица 2: Распределение слов по темам в тематической модели

Id темы	**Топ-10 слов с вероятностями**
1	0.019*"проект" + 0.014*"экономический" + 0.012*"реализация" + 0.011*"регион" + 0.011*"система" + 0.011*"создание" + 0.010*"деятельность" + 0.009*"программа" + 0.008*"инвестиционный" + 0.007*"предприятие"
2	0.067*"закон" + 0.035*"федеральный" + 0.032*"статья" + 0.029*"федерация" + 0.026*"изменение" + 0.021*"дума" + 0.020*"внесение" + 0.019*"проект" + 0.017*"пункт" + 0.013*"акт"
3	0.041*"комитет" + 0.029*"губернатор" + 0.026*"заместитель" + 0.022*"проект" + 0.019*"председатель" + 0.017*"ленин" + 0.016*"заседание" + 0.016*"экономический" + 0.015*"управление" + 0.014*"поручение"
4	0.117*"федерация" + 0.052*"орган" + 0.049*"федеральный" + 0.048*"субъект" + 0.035*"власть" + 0.025*"исполнительный" + 0.015*"президент" + 0.011*"совет" + 0.011*"россия" + 0.010*"округа"
5	0.017*"участие" + 0.012*"россия" + 0.011*"культура" + 0.011*"международный" + 0.010*"участник" + 0.009*"конкурс" + 0.009*"форум" + 0.008*"конференция" + 0.008*"проведение" + 0.008*"представитель"
6	0.056*"бюджет" + 0.031*"рубль" + 0.025*"средство" + 0.023*"субсидия" + 0.017*"сумма" + 0.016*"доход" + 0.016*"счёт" + 0.015*"расход" + 0.013*"налог" + 0.013*"финансовый"
7	0.026*"служба" + 0.022*"орган" + 0.021*"проверка" + 0.018*"управление" + 0.018*"контроль" + 0.017*"гражданский" + 0.013*"надзор" + 0.013*"нарушение" + 0.012*"служащий" + 0.012*"прокуратура"
8	0.041*"программа" + 0.031*"постановление" + 0.023*"год" + 0.021*"учреждение" + 0.020*"целевой" + 0.019*"проект" + 0.018*"мероприятие" + 0.017*"областной" + 0.016*"реализация" + 0.013*"бюджет"
9	0.025*"документ" + 0.024*"электронный" + 0.019*"информация" + 0.017*"информационный" + 0.015*"сведение" + 0.015*"адрес" + 0.014*"форма" + 0.014*"система" + 0.011*"наименование" + 0.011*"услуга"
10	0.037*"договор" + 0.034*"имущество" + 0.019*"собственность" + 0.019*"помещение" + 0.015*"общество" + 0.015*"сторона" + 0.014*"предприятие" + 0.013*"соглашение" + 0.012*"гоуп" + 0.012*"отношение"

1.4. Предсказание атрибута "Categories"

Вспомним, что атрибут "Categories" определен только для одной трети документов. Т.к. этот атрибут определяет тему документа, попробуем по распределениям тем, полученным из модели предсказать категорию документа. Для этого построим классификатор, которому в качестве множества описаний объектов передадим матрицу X, строка которой представляет из себя вектор вероятностей тем для документа, для которого известен атрибут "Categories". Множество значений этого атрибута для документов назначим целевой переменной Y (меткой класса). Документу в корпусе может соответствовать несколько кате-

горий. Но количество таких документов очень мало, поэтому исключим такие документы из выбранного множества объектов X. В качестве алгоритма многоклассовой классификации использовался метод опорных векторов (support vector machine). Была использована реализация метода из библиотеки scikit-learn [12].

При случайном разбиении множества объектов X на обучающее и тестовое множество в соотношении 60/40 точность предсказания целевой переменной Y на тестовом множестве составила 0.127. Было замечено, распределение значений целевой переменной в рассматриваемой выборке сильно неравномерно. Представителей некоторых классов достаточно мало и их шанс попасть в обучающее множество при таком разбиении так же не велик. При разбиении множества на обучающее и тестовое в том же соотношении так, чтобы в обучающее попали все уникальные значения целевой переменной, точность предсказания составила уже 0.373.

Было замечено, что 0.1 часть выборки является представителем класса, у которого значением атрибута "Categories" является значение с именем "Прочее". Именно для объектов этого класса классификатор ошибался больше всего. При исключении из множества объектов X элементов данного класса, точность предсказания на тестовом множестве составила 0.567.

Обученная таким образом модель классификатора далее использовалась для предсказания целевой переменной Y для объектов, не вошедших в обучающее множество. Таким образом мы смогли задать атрибут "Categories" для тех документов, для которых он был не известен.

2. Сокращение размерности

2.1. Способы перевода категориальных атрибутов в числовые

Основные алгоритмы кластеризации являются метрическими алгоритмами, которые разбивают выборку объектов на кластеры так, чтобы каждый кластер состоял из объектов, близких по некоторой метрике. Поэтому необходимо перевести категориальные атрибуты в числовые так, чтобы стандартные метрики , например Евклидово расстояние, в точности отражали близость объектов друг к другу.

Самый простой способ перевести категориальные атрибуты в числовые - это пронумеровать уникальные значения атрибута натуральными числами в каком-то порядке. Однако таким способом мы вносим лишнюю информацию в наши данные. Евклидова метрика в таком случае будет считать объекты с близкими значениями такого атрибута более похожими, что не верно, так так порядок нумерации был случайный и никак не обоснованный.

Самый популярный способ перевода категориальных атрибутов называется one hot encoding и заключается в следующем: атрибут j, принимающий n значений, заменяют на n признаков, принимающих значения 0 или 1, в зависимости от того, чему равно значение исходного признака j. Он оправдывает себя, если значение n не велико. При больших значениях n мы увеличиваем признаковое пространство на n измерений. Это может породить такую проблему, как "проклятие размерности" [15], которая для метрических классификаторов заключается в том, что чем больше размерность пространства тем больше похожи расстояния между двумя объектами. Следовательно тем сложнее разбить объекты на кластеры. Учитывая, что все атрибуты для наших документов категориальные и имеющие довольно большое количество уникальных значений, нам приходится отказаться от такого подхода.

2.2. Метод, основанный на решении задачи коммивояжера

Способ пронумеровать уникальные значения атрибутами числовыми значениями не так уж и плох, если попытаться задать правильный порядок нумерации атрибутов, т.е. расположить значения атрибутов в таком порядке, что чем ближе они находятся друг к другу, тем они более похожи. Но для этого нам необходимо задать функцию расстояния между значениями атрибута.

Во время анализа полученной тематической модели было выявлено, что значения всех атрибутов сильно коррелируют с темами, полученными из текстов. На рис. 3 для нескольких уникальных значений атрибутов представлена их зависимость от тем. Т.е. к примеру, оператор под номером 14 чаще создавал документы на тему под номером 9, чем на другие темы.

Рассмотрим один категориальный атрибут $x = x^1, ..., x^k$, где k - число уникальных значений атрибута. Введем вектор $p_i = p_i^1, ..., p_i^t$, где p_i^j - вероятность i-го значения атрибута в теме j. $i = \overline{1,k}$, $j = \overline{1,t}$. Имеем k вероятностных распределений, которые характеризуют k уникальных значений атрибута. Тогда для измерения расстояния между значениями атрибута воспользуемся расстоянием Дженсена-Шеннона:

$$JSD(P,Q) = \frac{1}{2}D_{KL}(P \parallel M) + \frac{1}{2}D_{KL}(Q \parallel M)$$

где $M = \frac{1}{2}(P+Q)$, $D_{KL}(Q \parallel M)$ - расстояние Кубака-Лейблера:

$$D_{KL}(P \parallel Q) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

Тогда расстояние $d(x^i, x^j)$ между i-тым j-тым значениями атрибута:

$$d(x^i, x^j) = JSD(p_i, p_j)$$

Теперь мы знаем расстояния между всеми уникальными значениями атрибута. Попробуем пронумеровать их в правильном порядке. Для

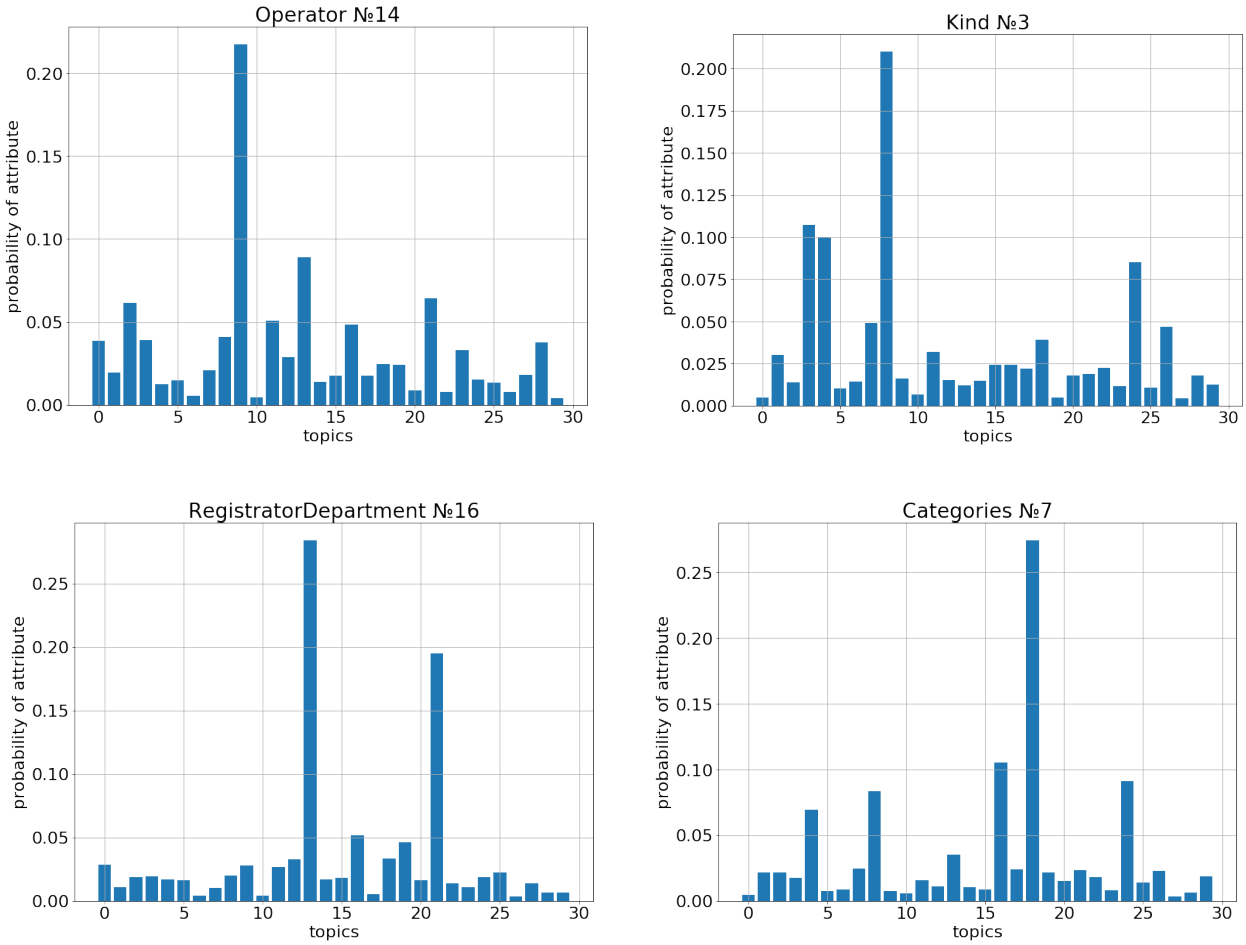

Рис. 3: Зависимость уникальных значений атрибутов от тем

этого воспользуемся постановкой и способами решения задачи коммивояжера. Задача является одной из самых известных задач комбинаторной оптимизации. Ее задачей является нахождение наиболее выгодного маршрута между объектами так, чтобы маршрут проходил по всем объектам ровно по одному разу и был замкнут. Критерий выгодности выбирается из условий задачи. Это может быть кратчайший путь, самый дешевый или самый быстрый. Обычно задачу представляют в виде графовой модели, где объекты являются вершинами графа, а ребра - взвешенными и отражают стоимость маршрута. Тогда поиск выгодного пути превращается в задачу нахождения в графе гамильтонова цикла минимального веса. Задача коммивояжера является NP-полной и решается различными приближенными алгоритмами.

Построим для каждого атрибута такой граф. Вершинами графа будут уникальные значения атрибута, а вес ребра (x^i, x^j) между верши-

нами x^i и x^j будет равен расстоянию между вершинами: $d(x^i, x^j) = JSD(p_i, p_j)$. Т.к. у нас есть расстояния между каждой парой значений атрибута, мы получим полный граф. Будем решать задачу коммивояжера на этом графе 2-приближенным методом, описанным в [10, p. 31], который обеспечивает решение задачи за полиномиальное время и нахождение маршрута максимум вдвое больше, чем оптимальный. Алгоритм состоит из следующих шагов:

1. Построить минимальное остовное дерево T.
2. Продублировать каждое ребро дерева T для получения эйлерова графа G.
3. Найти эйлеров обход в графе G.
4. Построить гамильтонов цикл путем исключения повторений вершин в эйлеровом обходе.

Полученный этим методом гамильтонов цикл будет не более чем в 2 раза больше оптимального (при условии, что веса в графе удовлетворяют условию неравенства треугольника). Получим гамильтонов путь, исключив в цикле ребро наибольшей длины. Т.к. в нашем графе расстояние между вершинами отражает похожесть значений атрибута друг на друга, то в полученном гамильтоновом пути вершины будут расположены так, что наиболее похожие будут располагаться рядом и являться соседями.

Пришло время присвоить вершинам числовые значения. Первой вершине присвоим 0, а всем последующим присвоим сумму из значения предыдущей вершины и длине пути до нее.

Решим эту задачу для всех атрибутов документа. Далее необходимо произвести нормализацию полученных значений атрибута. Сделаем это методом MinMax Scaling, который переносит все точки на отрезок $[0, 1]$:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Для построения графа и применения алгоритмов построения остовного дерева и нахождения эйлерова цикла использовалась Python-библиотека NetworkX [7].

Посмотрим на результат работы метода на примере атрибута "Categories". Как упоминалось ранее у этого атрибута есть достаточное количество дубликатов. Логично предположить, что в полученном пути дубликаты должны оказаться соседями, и это действительно наблюдалось. Приведу несколько значений атрибута, которые после применения алгоритма расположились друг за другом:

- 'Жилищно-коммунальное хозяйство', 'КОММУНАЛЬНОЕ ХОЗЯЙСТВО (ЖИЛЬЕ)'
- 'ФИНАНСЫ', 'Финансы', 'Бюджет, кредит, цены'
- 'Статистика', 'Статданные'
- 'Строительство', 'ДОРОЖНОЕ СТРОИТЕЛЬСТВО', 'Дорожное хозяйство'
- 'Общественные связи', 'Общественные инициативы', 'Общественные организации', '35.Общественные организации и партии'

Таким образом, мы перевели категориальные атрибуты в числовые, не сильно увеличив размерность данных.

2.3. Многомерное шкалирование

Попробуем перевести категориальные атрибуты в числовые методом многомерного шкалирования (multidimensional scaling). Это метод нелинейного понижения размерности, цель которого для всех объектов из обучающей выборки, для которых известны попарные расстояния, построить его признаковое описание в маломерном Евклидовом пространстве. При этом евклидовы расстояния между объектами в новом пространстве должны как можно точнее приближать исходные расстояния между объектами. Метод удобен для задач, в которых об объектах

известны только попарные расстояния, а признаковые описания не известны.

Пусть $x_1, ..., x_l$ - объекты в исходном пространстве, а $\widetilde{x}_1, ..., \widetilde{x}_l$ - объекты в маломерном пространстве. Расстояния в исходном пространстве обозначим, как $d_{ij} = \rho(x_i, x_j)$, а расстояния в маломерном пространстве $\widetilde{d_{ij}} = \| \widetilde{x}_i - \widetilde{x}_j \|$, которые измеряются с помощью евклидовой метрики. Задача многомерного шкалирования - минимизировать функционал, который для всех пар объектов измеряет квадратичное отклонение между расстояниями в исходном и маломерном пространствах:

$$\sum_{i<j}^{l} (\| \widetilde{x}_i - \widetilde{x}_j \| - d_{ij})^2 \to \min_{\widetilde{x_1},...,\widetilde{x_l}}$$

В качестве объектов будем рассматривать уникальные значения атрибута x: $x^1, ..., x^k$. В качестве функции расстояния воспользуемся использованным ранее расстоянием Дженсена-Шеннона:
$d(x^i, x^j) = JSD(p_i, p_j)$, где p_i - вектор распределения вероятностей i-го значения атрибута по темам. Для сравнения этого метода с рассмотренным ранее методом, основанным на решении задачи коммивояжера, размерность маломерного пространства примем за 1.

Для решения задачи использовалась реализация алгоритма многомерного шкалирования из Python-библиотеки scikit-learn [12].

Аналогично методу, основанному на задачи коммивояжера, посмотрим на новые числовые значения для атрибута "Categories". Полученные результаты так же показывали, что похожие категории располагались последовательно:

- 'Строительство', '02.Строительство', 'Архитектура, градостроительство'
- '21.Экология и охрана природы', 'Экология и недропользование'

Попробуем теперь каждый атрибут перевести в евклидово пространство размерности 2. Так мы увеличим размерность признаков документа в 2 раза, но, возможно, в них сохранится больше полезной информации. Опять же проверим результаты на атрибуте "Categories". Т.к.

теперь уникальное значение атрибута представляется в виде вектора размерности 2, отобразим категории на плоскости и посмотрим на их взаимное расположение. На рис. 4 представлены 2 области на полученном графике. Видно, что похожие категории опять расположились рядом.

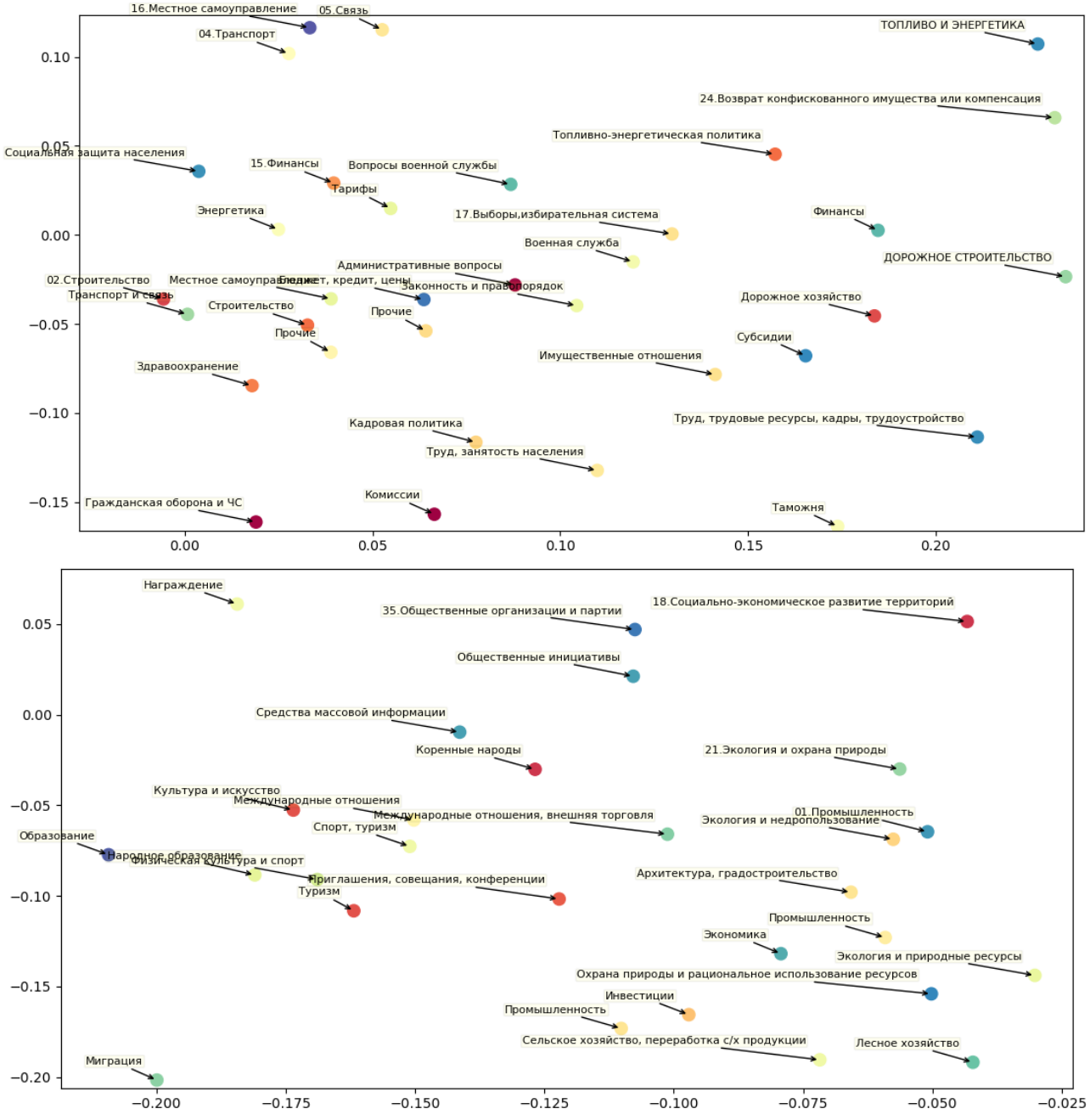

Рис. 4: Взаимное расположение значений атрибута ”Categories” после сокращения размерности методом многомерного шкалирования

3. Кластеризация

3.1. Описание входных данных и введение метрики

Вспомним, что исходные данные представляли из себя набор из 131214 объектов, которые описывались восьмью атрибутами. Атрибут "Sender", который характеризует отправителя мы не рассматриваем из-за большого числа уникальных для него значений. Недостающие значения для атрибута "Categories" были предсказаны, как описано в главе 1.4. При помощи методов перевода категориальных атрибутов в числовые, описанных в главе 2, мы получили 3 различных числовых описаний объектов в евклидовом пространстве. Метод, основанный на решении задачи коммивояжера, оставил размерность признакового описания объектов равной семи, а при помощи метода многомерного шкалирования были получены признаковые описания объектов размерности 7 и 14.

Большинство алгоритмов кластеризации являются метрическими и требуют введения метрики расстояния между исследуемыми объектами. Т.к. мы привели категориальные атрибуты в числовые, представленные в евклидовом пространстве, мы можем использовать в качестве метрики расстояния между объектами евклидову метрику.

3.2. Алгоритмы кластеризации

Были рассмотрены два алгоритма кластеризации. Первый, наиболее популярный, алгоритм k-средних (k-means). Идея метода заключается в том, что на каждой итерации для каждого кластера перевычисляется центр масс, а объекты перераспределяются по тем кластерам, чей центр масс оказался на этой итерации ближе. Цель алгоритма найти такую функцию кластеризации $C : x \to 1, ..., K$, которая минимизирует суммарную вариацию внутри кластеров:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(x_i)=k} \sum_{j:C(x_j)=k} \rho(x_i, x_j)$$

где K - число кластеров.

Вторым алгоритмом был рассмотрен агломерационный иерархический алгоритм кластеризации (agglomerative clustering). Стратегия метода следующая: сначала каждый объект представляет отдельный кластер. Затем запускается процесс слияний, при котором на каждой итерации два наиболее близких кластера по заданной метрике объединяются в один кластер. Метрикой расстояния между кластерами было выбрано значение среднего расстояния между всеми парами объектов двух кластеров:

$$\rho(N, M) = \frac{1}{|N||M|} \sum_{n \in N} \sum_{m \in M} \rho(n, m)$$

, где N и M - пара кластеров. Процесс объединения кластеров прекращается при достижении необходимого числа кластеров.

Использовались реализации алгоритмов из Python-библиотеки scikit-learn [12].

4. Эксперименты и оценки результатов

Каждый алгоритм кластеризации был запущен на всех 3 наборах данных, полученных после перевода категориальных атрибутов в числовые методами, описанными в главе 2. В каждом эксперименте данные разбивались на 50 кластеров.

Для сравнения полученных результатов друг с другом необходимо их как-то оценить. Стандартные оценки, основанные на расстояниях между элементами кластеров, такие как минимум среднего внутрикластерного расстояния или минимум среднего межкластерного расстояния, нам не совсем подходят, т.к. для нас такая минимизация не очень важна. Чтобы в последствии пользоваться стратегией "Разделяй и властвуй", нам будет более важным тот факт, что документы с одинаковым значением некоторого атрибута оказались в одном кластере. Поэтому для оценки качества кластеров была выбрана следующая стратегия.

Рассмотрим вектор $p_i = p_i^1, ..., p_i^t$, где p_i^j - вероятность i-го значения атрибута в кластере j. Нам выгодно, чтобы только небольшая часть элементов вектора p_i имела наибольшие значения, а остальные были близки к нулю. Численно мы можем посчитать такую выгоду при помощи энтропии:

$$H(p_i) = -\sum_{j=1}^{t} p_i^j \log p_i^j$$

которая покажет меру неопределенности появления уникального значения атрибута в той или иной теме. Чем меньше значение энтропии, тем больше мы уверены в том, что такое значение атрибута характерно для малого числа кластеров. Для каждого атрибута посчитаем среднее значение энтропии по всем уникальным значениям этого атрибута:

$$H_{avr}(x) = \frac{1}{k} \sum_{i=1}^{k} H(p_i)$$

где x - атрибут, у которого t уникальных значений. И сравним полученные результаты по этому среднему значению энтропии.

Таблица 3: Средние значение энтропии для атрибутов при кластеризации методом k-средних

x	t	$H_{avr}(x)$, $x \in$ **tsp**	$H_{avr}(x)$, $x \in$ **mds1**	$H_{avr}(x)$, $x \in$ **mds2**
AccessType	3	2.905	1.349	2.678
Kind	19	1.631	0.918	1.134
Operator	145	0.411	0.220	0.423
Registrator	158	0.422	0.263	0.426
RegistratorDepartment	66	0.405	0.654	0.461
Recipients	634	0.333	0.311	0.253
Categories	157	1.985	0.764	1.289
Topics	30	4.613	3.547	4.054

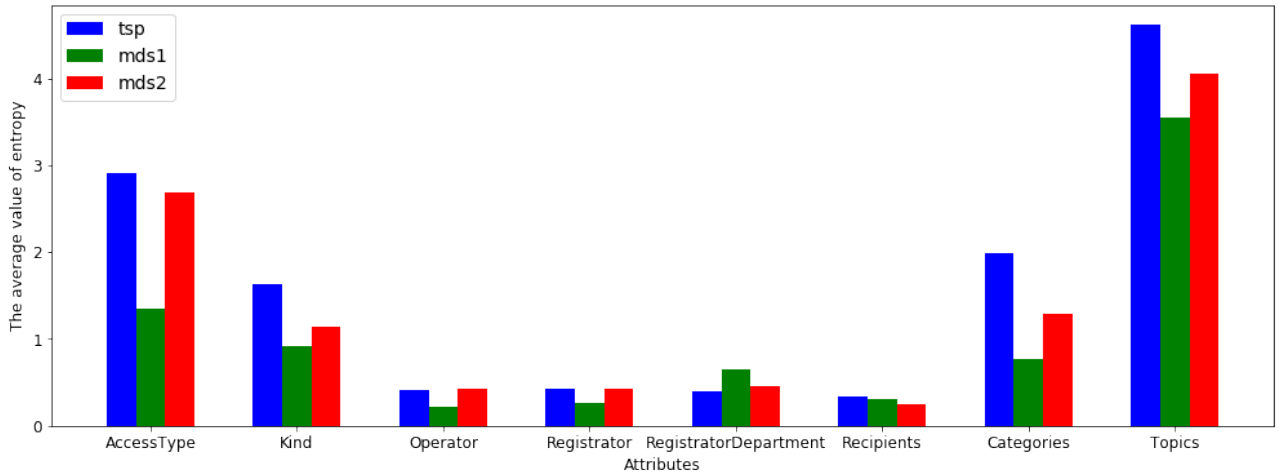

Рис. 5: Средние значение энтропии для атрибутов при кластеризации методом k-средних

Посмотрим, как методы кластеризации отработали на трех наборах данных: tsp - набор, полученный после перевода категориальных атрибутов в числовые методом, основанным на решении задачи коммивояжера; mds1 - методом многомерного шкалирования в одномерное евклидово пространство; mds2 - в двумерное. Результаты работы метода k-средних представлены в Таблице 3, а отношение результатов на разных наборах данных друг к другу хорошо видно на рис. 5. Результаты работы иерархического алгоритма представлены в Таблице 4 и на рис. 6.

На графиках видно, что лучше всего по кластерам разбились атрибуты Оператор, Регистратор, Подразделение регистрации и Получате-

Таблица 4: Средние значение энтропии для атрибутов при кластеризации иерархическим методом

x	t	$H_{avr}(x)$, $x \in$ **tsp**	$H_{avr}(x)$, $x \in$ **mds1**	$H_{avr}(x)$, $x \in$ **mds2**
AccessType	3	2.903	0.859	2.511
Kind	19	1.717	0.693	1.085
Operator	145	0.418	0.408	0.290
Registrator	158	0.436	0.442	0.366
RegistratorDepartment	66	0.444	0.752	0.487
Recipients	634	0.348	0.429	0.307
Categories	157	1.983	0.630	1.098
Topics	30	4.528	1.727	3.865

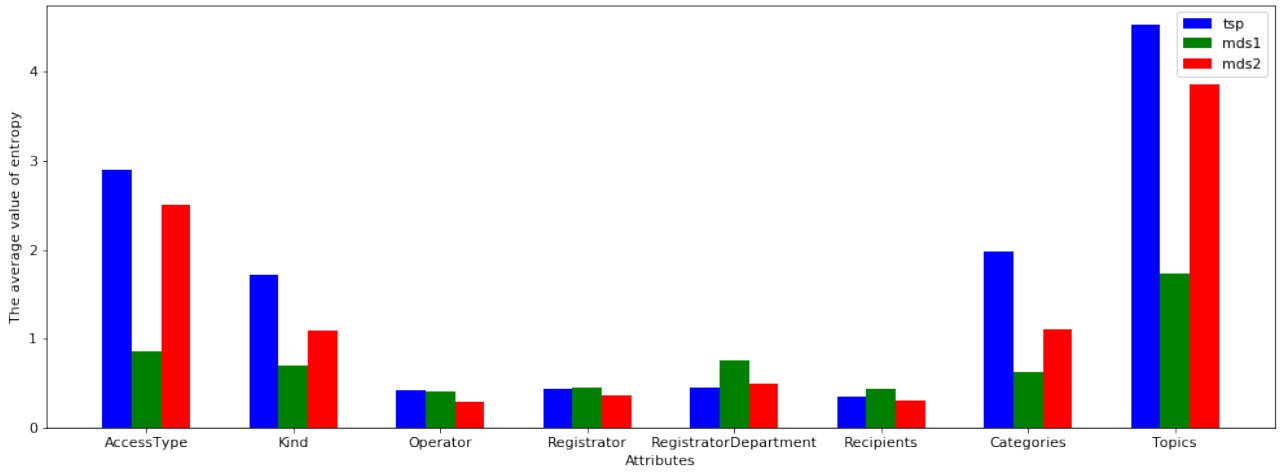

Рис. 6: Средние значение энтропии для атрибутов при кластеризации иерархическим методом

ли, т.к. у них на всех экспериментах низкая средняя энтропия. Кроме того, на наборе mds1 алгоритмы в среднем по атрибутам показывают лучшие результаты. Сравним методы кластеризации на этом наборе. Результаты представлены в Таблице 5 и на рис. 7. В среднем по атрибутам иерархический алгоритм отработал лучше алгоритма k-средних.

Однако все эти выводы дают только примерную оценку рассмотренных алгоритмов. При применении же стратегии ”Разделяй и властвуй” один и тот же алгоритм кластеризации для одних задач может дать хорошие результаты, а для других плохие. Поэтому выбор алгоритма класреризации лучше делать на этапе решения определенной задачи, сравнивая, какой из методов улучшил ее решение.

Таблица 5: Сравнение двух методов кластеризации на наборе mds1

	K-means	**Agg_clustering**
AccessType	1.349	0.859
Kind	0.918	0.693
Operator	0.220	0.408
Registrator	0.263	0.442
RegistratorDepartment	0.654	0.752
Recipients	0.311	0.429
Categories	0.764	0.630
Topics	3.547	1.727

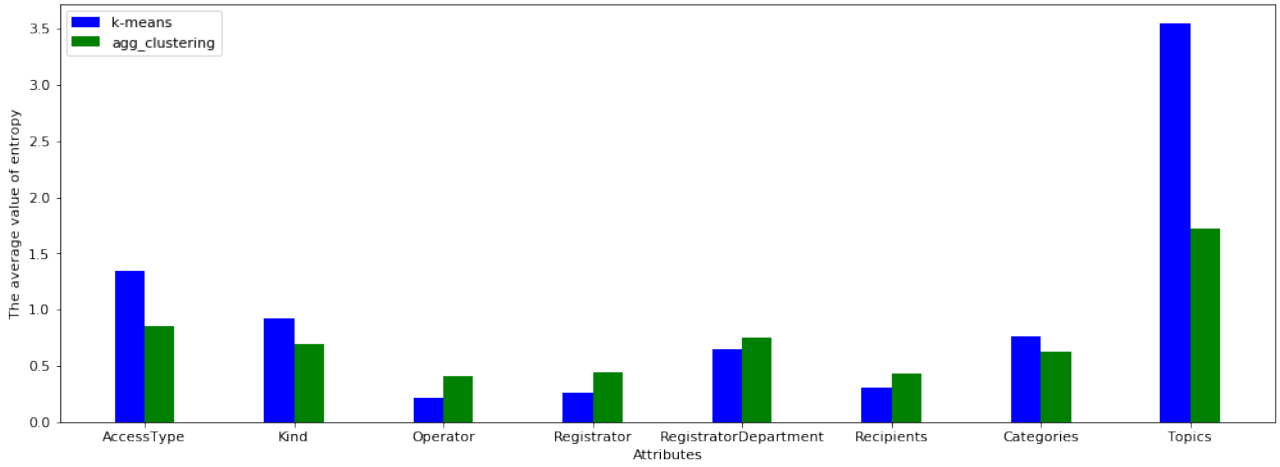

Рис. 7: Сравнение двух методов кластеризации на наборе mds1

Заключение

В данной работе была рассмотрена задача кластеризации документов в системе электронного документооборота. Т.к. документы в системе содержат не только текстовую информацию, но и описываются некоторым набором атрибутов, необходимо было при кластеризации учитывать все эти параметры.

Из текстовой информации были получены распределения тем для каждого документа. Полученные распределения позволили предсказать важный атрибут для документов "Категория". Он был известен только для трети документов. Точность предсказания составила 0.567, что является хорошим результатом, учитывая, что у атрибута "Категория" среди его уникальных значений, которых всего 157, было обнаружено достаточно много дубликатов.

Были рассмотрены различные способы перевода категориальных атрибутов в числовые. Введение метрики, которая учитывает близость значений атрибутов исходя из распределений тем, полученных из тематической модели, позволило легко применить следующие методы. Был предложен метод, основанный на решении задачи коммивояжера, который не плохо себя показал. Для этой же задачи был адаптирован метод многомерного шкалирования, который обычно применяется для понижения размерности данных.

Были протестированы два алгоритма кластеризации: k-средних и агломерационный иерархический алгоритм. На вход алгоритмам подавалась матрица признаковых описаний документов, в которой категориальные признаки были преобразованы в числовые тремя предложенными методами.

В результате экспериментов показал себя наилучшим образом иерархический алгоритм, который получил на вход данные, значения атрибутов которых были преобразованы алгоритмом многомерного шкалирования в вектор размерности 1. Естественно он показал себя хорошо только на той метрике (среднем значении энтропии), которую мы посчитали для всех экспериментов. При применении этих алгоритмов в

контексте других задач, метрика может быть выбрана совершенно другая. Если же предложенные алгоритмы применять в стратегии "Разделяй и властвуй", то качество алгоритмов будет определять коэффициент, показывающий, насколько использование стратегии с этим методом улучшило исходный алгоритм.

Учитывая объемы накопленных документов в системах документооборота и их постоянное пополнение, перед применением любых алгоритмов анализа данных будет полезным умение разбивать документы на кластеры.

Список литературы

[1] Anil K. Jain Richard C. Dubes. Algorithms for Clustering Data. — Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1988. — ACM Digital Library : http://dl.acm.org/citation.cfm?id=42779.

[2] David M. Blei Andrew Y. Ng Michael I. Jordan. Latent Dirichlet Allocation. — Journal of Machine Learning Research, 2003. — http://www.jmlr.org/papers/v3/blei03a.html.

[3] Digital Design. — 2017. — URL: `http://digdes.ru/` (дата обращения: 10.05.2017).

[4] Gensim. Topic modelling for humans. — 2017. — URL: `https://radimrehurek.com/gensim/` (online; accessed: 10.05.2017).

[5] Heinrich Gregor. Parameter estimation for text analysis. — University of Leipzig, Tech. Rep., 2008. — https://faculty.cs.byu.edu/ ringger/CS601R/papers/Heinrich-GibbsLDA.pdf.

[6] Natural Language Toolkit. — 2017. — URL: `http://www.nltk.org/` (online; accessed: 10.05.2017).

[7] NetworkX. — 2017. — URL: `https://networkx.github.io/` (online; accessed: 10.05.2017).

[8] Sievert C. Shirley K. E. LDAvis: A method for visualizing and interpreting topics. — Proceedings of the workshop on interactive language learning, visualization, and interfaces, 2014. — http://www.aclweb.org/website/old_anthology/W/W14/W14-31.pdf.

[9] Tesseract OCR // github.com. — 2017. — URL: `https://github.com/tesseract-ocr/` (online; accessed: 10.10.2016).

[10] V. Vazirani V. Approximation algorithms. — Springer Science & Business Media, 2013.

[11] pyLDAvis // github.com. — 2017. — URL: https://github.com/bmabey/pyLDAvis (online; accessed: 10.05.2017).

[12] scikit-learn. — 2017. — URL: http://scikit-learn.org (online; accessed: 10.05.2017).

[13] Кластеризация // machinelearning.ru. — 2011. — URL: http://www.machinelearning.ru/wiki/index.php?title=Кластеризация (дата обращения: 01.09.2016).

[14] Морфологический анализатор pymorphy2. — 2017. — URL: http://pymorphy2.readthedocs.io (дата обращения: 10.05.2017).

[15] Проклятие размерности // machinelearning.ru. — 2017. — URL: http://www.machinelearning.ru/wiki/index.php?title=проклятие_размерности (дата обращения: 10.05.2017).

[16] СЭД Docsvision. — 2017. — URL: http://www.docsvision.com/ (дата обращения: 01.05.2017).